Reversing Protected Minutiae Vicinities

Koen Simoens, Chi-Ming Chang, and Bart Preneel

Department of Electrical Engineering - ESAT/COSIC
Katholieke Universiteit Leuven and IBBT, Belgium

***Abstract*—In this paper we analyze the template protection method for minutiae-based fingerprint biometrics as proposed by Yang and Busch (BTAS 2009). This method is based on a self-aligning and non-invertible parameterized transformation that is applied to minutiae vicinities. Two attack strategies to invert protected vicinities are presented and we point out that the attack complexity is well below the estimates in the original paper. Improvements to increase the complexity are suggested.**

I. Introduction

Security and privacy issues associated with the increasing deployment of biometric systems have widely been recognized and addressed by the scientific community [1] and policy makers [2]. To deal with these issues several approaches have been explored to protect biometric data at the template level. Since traditional encryption schemes cannot deal with the noise that is inherent to biometrics, new methods have been invented that are capable of hiding the biometric data, while at the same preserving the biometric identification and verification functionality. In addition, these methods introduce diversification to make protected templates unlinkable and renewable.

Over the last decade, a vast amount of papers have proposed schemes to protect biometric templates [3]–[12], and minutiae-based fingerprint templates in particular [13]–[21]. Many of the proposed template protection methods can be categorized as noise-tolerant algorithms. They mostly use error-correcting codes to compensate for the noise. One of the most popular techniques is the fuzzy commitment scheme [4], which has been used for many different types of biometrics [9], [22]. The application of such schemes is limited since they can only deal with discrete, i.e., quantized or binary, data. Moreover, they tend to reveal a substantial amount of biometric data [23]. Another category of methods deals with continuous-source biometrics and compensates for the noise during quantization [7]. Despite the good properties they possess, most fuzzy schemes suffer from the limitation that their inputs have to be of fixed length, which is not the case for a minutiae vector extracted from a fingerprint. Several attempts have been made, though, to overcome this problem [14], [24], [25].

In 2001 Ratha et al. [5] introduced the concept of cancelable biometrics. Although the cancelability referred mainly to the fact that a protected template can be revoked and renewed, without being linkable, the term is now commonly associated with a protection method that applies a non-invertible, parameterized transformation in the signal or feature domain. The benefits of this approach are that the non-invertible transformation can be combined with existing (unprotected) comparison algorithms and that they suffer less from the performance degradation that is typically observed when template protection is introduced. Moreover, cancelable biometrics can deal with more input types. Global transformations for fingerprint biometrics have been proposed by Ratha et al. [20]. However, the linkability and non-invertibility of their transformations have recently been challenged [26].

One of the drawbacks of a global transformation is that the templates need to be pre-aligned. To overcome this problem, Yang and Busch [21] have presented a new method for self-alignment and protection of fingerprint templates. Each template is decomposed in a set of minutiae vicinities. To each of these vicinities the self-alignment and a non-invertible transformation are applied. Comparison of two protected templates is done vicinity-wise. The main objective of this work is to analyze the security of the method proposed by Yang and Busch. We will evaluate the vicinity transformation and revise its security analysis. Two attack strategies are elaborated and we will show that it is possible to reverse protected minutiae vicinities at a high level of accuracy.

The paper is organized as follows. The main concepts of cancelable biometrics for minutiae-based fingerprint templates are introduced in Section II, along with some notation, and we show how these concepts are used in the scheme proposed by Yang and Busch [21]. After the analysis of the scheme, two attack strategies are described and evaluated in Section III. We discuss the results and evaluate how the properties of the scheme lead to the reversibility of protected minutiae vicinities. Section IV concludes this paper.

II. Cancelable Minutiae Templates

For the purpose of this paper we will briefly reiterate the main concepts of cancelable biometrics and introduce some notation for fingerprint minutiae templates that is needed for the rest of this paper. We will follow to a large extent the notation presented in [21].

A. Minutiae Vicinities

A fingerprint minutiae template T is defined as a set of points m_i, the minutiae, which are represented as triplets indicating their coordinates and orientation in the fingerprint:

$$T = \{m_i\}_{i=1..N} = \{(x_i, y_i, \theta_i)\}_{i=1..N}\,.$$

This work was sponsored in part by the EU project TURBINE, which is funded by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement nb. ICT-2007-216339.

Alternatively, the minutiae template can be represented as a set of minutiae vicinities. A minutiae vicinity is defined as a set of $(M+1)$ minutia points, i.e., a minutia point m_i and its M closest neighbors c_{ij}:

$$T^V = \{V_i\}_{i=1..N} = \{\, \{m_i, c_{ij}\}_{j=1..M} \,\}_{i=1..N} .$$

B. Cancelable Biometrics

The idea of a cancelable biometrics algorithm as proposed by Ratha et al. [5] is to apply a non-invertible (keyed) transformation F in the signal or feature domain in such a way that comparison algorithms that exist for unprotected templates can also be used to compare protected templates. In the context of fingerprint recognition, a minutia matcher for unprotected templates T and T' can be used to compare the protected variants $T_P = F(T)$ and $T'_P = F(T')$. The protected template is again a set of minutiae points, i.e.,

$$T_P = \{m'_i\}_{i=1..K}$$

and K and N are not necessarily equal. When we consider vicinities we will assume that each vicinity is protected individually:

$$T_P^V = \{V_{i,P}\}_{i=1..N} = \{F(V_i)\}_{i=1..N} .$$

Similarly the number of minutiae in V_i does not have to be the same as the number of minutiae in $V_{i,P}$.

Often the transformation depends on a secret key K. To make it explicit we write $T_P = F(T, K)$. The key can, in theory, be the same for all users but will most likely be different due to re-enrollments. In addition to an extra level of security, the key provides a way to diversify the templates.

The security properties of a cancelable fingerprint transformation F were specified in [20] in terms of a matching algorithm. For example, two templates that are produced with a different transformation are unlinkable if they are not classified as related by the matching algorithm. The property of main interest for this paper is the irreversibility, which states that a template T should not be recoverable from its protected variant $T_P = F(T)$, i.e., the inverse of F must not exist. Assuming that a key is used to perform the transformation we distinguish between the following two forms of irreversibility[1].

- A transformation F provides *weak irreversibility* if for any adversary who knows the transformation F it is infeasible to recover T from $T_P = F(T, K)$.
- A transformation F provides *strong irreversibility* if for any adversary who knows the transformation F *and the key* K it is infeasible to recover T from $T_P = F(T, K)$.

C. Minutiae Vicinity Method

At BTAS 2009 a method for protecting fingerprint templates based on minutiae vicinities was proposed by Yang and Busch [21]. A minutiae template T is transformed into a vicinity template T^V and alignment and protection are applied to each vicinity individually. The self-alignment avoids the unreliable pre-alignment of the full minutiae template based on core points. The method is explained in this section.

1) Self-Alignment: Recall that a vicinity consists of a leading minutia m_i and its M closest neighbors c_{ij}. In each vicinity L orientations are chosen, along which the vicinity will be aligned (e.g. Fig. 1). An orientation is defined by an orientation vector, i.e., an ordered pair of minutiae, e.g., $\overrightarrow{m_i c_{i1}}$. The vicinity is translated and rotated in such a way that the direction of the vector $\overrightarrow{m_i c_{i1}}$ is used as the new x-axis and the middle point as the new origin. The result of the alignment is shown in Fig. 2 (the aligned vicinity consists of the points m'_i, c'_{i1}, c'_{i2} and c'_{i3}).

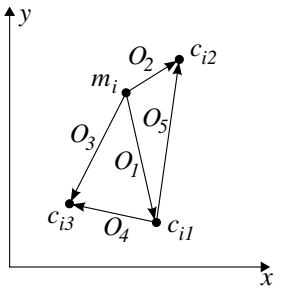


Fig. 1. Orientations O_1 to O_5 in vicinity $V_i = \{m_i, c_{i1}, c_{i2}, c_{i3}\}$ [21].

For L orientations, there are L aligned groups and $(M-1)$ points per group, i.e., not including the points that determine the orientation. Only L of the $2\binom{M+1}{2}$ possible orientations are used in order to reduce the template size. In this paper it is assumed that an attacker knows how the orientations are chosen, e.g., the algorithm may choose the L longest orientation vectors. This is a reasonable assumption as the security of the algorithm should not depend on some obscure techniques. In the original paper $M = 3$ and $L = 5$.

2) Offsetting: The protection is based on a parameterized offsetting of the minutiae in the self-aligned vicinity. The minutia points that defined the new x-axis are ignored. Offsets are added to the coordinates of the self-aligned minutiae in polar direction and their corresponding perpendicular direction, as illustrated in Fig. 2.

For each J'_j in the self-aligned vicinity that did not determine the orientation vector the coordinates are offset as

$$\begin{cases} J_{aj}(x) = J'_j(x) + \Delta x_l \cdot cos(\varphi) + \Delta y_l \cdot cos(\varphi + \frac{\pi}{2}) \\ J_{aj}(y) = J'_j(y) + \Delta x_l \cdot sin(\varphi) + \Delta y_l \cdot sin(\varphi + \frac{\pi}{2}) \end{cases} \quad (1)$$

where the polar direction φ is defined by the aligned minutia direction

$$\varphi = tan^{-1}\left(\frac{J'_j(y)}{J'_j(x)}\right) . \quad (2)$$

The offsets Δx_l and Δy_l are different for each orientation and are calculated as

$$\begin{cases} \Delta x_l = D_l \cdot dx_l + D_{max} \cdot sign(dx_l) \\ \Delta y_l = D_l \cdot dy_l + D_{max} \cdot sign(dy_l) \end{cases} . \quad (3)$$

The parameters dx_l and dy_l are randomly generated during enrollment and stored in a key table K, which contains L

[1]We opt for this terminology, in analogy of the weak and strong biometric privacy requirements for key generation as defined by Ballard et al. [27].

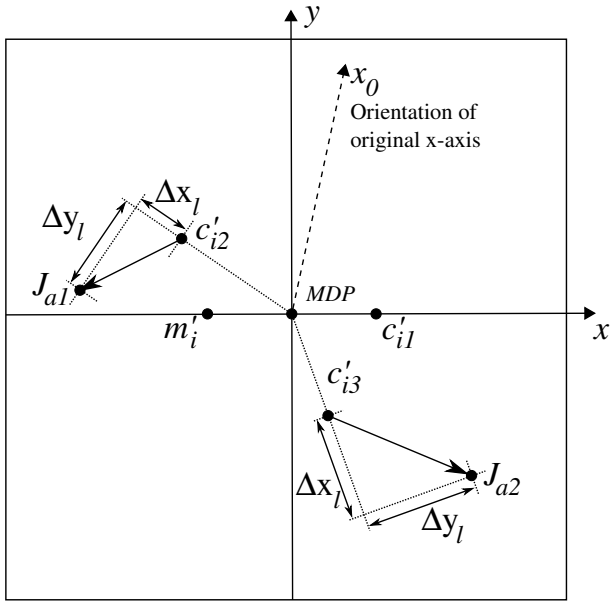


Fig. 2. Self-aligned and offset vicinity after alignment along $O_1 = \overrightarrow{m_i c_{i1}}$. The minutiae that determined the orientation vector are ignored while the remaining self-aligned minutiae c'_{i1} and c'_{i2} are offset to J_{a1} and J_{a2} [21].

entries, one for each orientation. The values of dx and dy are in the range $[-1, 1]$. The parameter D_l is the length of the orientation vector that was used for the alignment along orientation O_l and $D_{max} = \max\{D_l : l = 1..L\}$ is the length of the longest of the L orientation vectors that are used for alignment. The L groups of self-aligned and offset points are then superimposed. The result is the full protected vicinity, which contains $L*(M–1)$ protected minutia points. Comparison of protected templates is done on a vicinity basis. The final mathcing score is computed as the number of matching vicinities.

3) Attack Complexity: The security analysis in [21] estimates that the complexity of inverting a protected template by guessing ten of the original minutiae is approximately 2^{85}, i.e., the number of possible combinations of ten points given the size of the template and a minimum distance (the ridge width) between any pair of minutiae. Due to the vicinity approach, however, the complexity is reduced to the complexity of inverting protected vicinities. The complexity of a brute-force attack on a vicinity by guessing the location of the M neighboring minutiae depends on the dynamic range R_o of the coordinates in the original (unprotected) vicinity. The range R_o from the leading minutia to its neighbors is estimated to be at most 200. For a ridge width $W_R = 10$ and $M = 3$, the complexity is estimated at $\binom{\lfloor \pi R_o^2 / W_R^2 \rfloor}{M} \approx 2^{28}$. This complexity is considerably low, but it is assumed that the attacker knows the key, i.e., the random offset table, or has access to some functionality that allows him to regenerate a protected vicinity on input of a guessed vicinity.

The complexity of an inversion attack based on the protection algorithm and the information in the protected template was estimated at 2^{67} given that the attacker knows the key. Without the key, the complexity is estimated at 2^{120}. In the next section, we will demonstrate that the complexity can be reduced to 2^{17} for an attacker who knows the key. Assuming that the attacker does not know the key is a strong and unrealistic assumption, since anyone, e.g., an insider, can be an attacker.

III. Attacks

In this section we will present two attack strategies that challenge the strong irreversibility of the scheme proposed in [21]. The protected template is the aggregate of N protected vicinities and thus the goal in each strategy is to invert a protected minutiae vicinity. Our attack model assumes that the adversary knows the transformation function, how to choose the orientations and which entry in the key table (random offsets) is used for which orientation.

From now on we will refer to the original vicinity in self-aligned form. Each original vicinity consists of $(M + 1)$ points, which are consequently transformed to a set of L superimposed sets of $(M-1)$ points. Given the parameters in [21], an original vicinity consists of four minutia points whereas the protected vicinities contain ten points. If a correct pair is chosen in the protected vicinity and if the corresponding offsets are known, then this pair can be shifted back to its original (self-aligned) position. Because the points that determine the orientation vector are located on the x-axis and symmetrically around the origin, these can be recovered as well if the length of the orientation vector is known. The problem of inverting the protected vicinity thus boils down to finding the offsets and the length of the orientation vector.

Recall from (1) that the offsets Δx_l and Δy_l are given by

$$\begin{cases} \Delta x_l = D_l \cdot dx_l + D_{max} \cdot sign(dx_l) \\ \Delta y_l = D_l \cdot dy_l + D_{max} \cdot sign(dy_l) \end{cases}.$$

Finding these offsets, given the key (dx_l, dy_l), is equivalent to finding D_l, the length of the current orientation vector, and D_{max}, the length of the longest orientation vector. The value of D_l also reveals the aligned coordinates of the minutiae that defined the orientation vector, i.e., $(-D_l/2, 0)$ and $(D_l/2, 0)$.

A. Maximum Orientation Length (Dmax) Attack

1) Attack Strategy: In the alignment and offsetting along the longest orientation vector, we have that $D_l = D_{max}$. The offsets can be written as

$$\begin{cases} \Delta x_l = D_{max} \cdot sign(dx_l) \cdot (1 + |dx_l|) \\ \Delta y_l = D_{max} \cdot sign(dy_l) \cdot (1 + |dy_l|) \end{cases} \quad (4)$$

and the ratio between Δx_l and Δy_l is

$$\frac{\Delta x_l}{\Delta y_l} = \frac{sign(dx_l) \cdot (1 + |dx_l|)}{sign(dy_l) \cdot (1 + |dy_l|)}. \quad (5)$$

Guessing the offsets is reduced to guessing the value of D_{max}. Given the correct subset of $(M-1)$ points in the protected vicinity and the key (dx_l, dy_l), the attack strategy is to iterate over the range of D_{max} and to compute the offsets Δx_l and Δy_l in each iteration. The potential original vicinity is reconstructed by inverting the offsets and by positioning the

minutiae of the orientation vector on the x-axis. To validate the guess of D_{max}, the protected vicinity is regenerated from the guessed original vicinity and compared with the given protected vicinity. A simple distance error function is used to validate the guess.

Unfortunately, the subset of $(M-1)$ protected minutiae and the key (dx_l, dy_l) corresponding to the longest orientation vector are not known. Moreover, there is only one orientation, and thus only one entry in the key table, for which it holds that $D_l = D_{max}$. Therefore, we will exhaustively search over all possible subsets of size $(M-1)$ in the protected vicinity and over all key entries. From the combination (minutiae pair, key entry) that yields the smallest distance error, the original vicinity will be recovered. Given the parameters in [21] ($M = 3$, $L = 5$ and $D_{max} \in [0, 2 * R_o]$ for $R_o < 200$) the worst-case complexity of the attack is

$$\binom{L*(M-1)}{M-1} \cdot L \cdot 2 * R_o = \binom{10}{2} \cdot 5 \cdot 400 \approx 2^{17}. \quad (6)$$

2) Attack Algorithm: For practical reasons, we guess the phase difference θ between the minutia points before and after the offset, instead of guessing D_{max} or Δx_l directly. The phase iterates from $-\pi/2$ to $\pi/2$ for a given resolution θ_{res}, i.e., the phase estimate is increased with $(\pi/2)/\theta_{res}$ in. each iteration. It is possible that a guess yields an impossible result, e.g., if after reversing the offsets the recovered minutiae fall on the opposite side of the original protected minutiae. A detailed description of the attack algorithm is given in Appendix (Algorithm 1).

3) Experimental Results: To simulate the attack algorithm a vicinity V of four minutia positions is randomly generated on a frame of 200×200 pixels. From V a protected vicinity V_P is generated on which the Dmax attack algorithm is run. Step 2 of the algorithm, i.e., the search for the best θ given a minutiae pair (m_i, m_j) and key entry (dx_l, dy_l) is shown in Fig. 3(a). For point m_i, all possible pre-offset positions m_i' form an elliptic trajectory that passes through m_i. The trajectory for m_j is incomplete because some $(\Delta x_l, \Delta y_l)$ move m_j to the invalid region. If the minutiae pair that was selected in V_P is correct for the given key entry, then the two trajectories will pass closely to the two pre-offset minutiae that were aligned along the D_{max} orientation.

The search is performed over all pairs in V_P and for all key entries in K. The best guess has a protected vicinity that is closest to the given vicinity. The distance errors for the best offset guesses, based on a resolution $\theta_{res} = 30$, for all pairs and all key entries are shown in Fig. 4. The smallest error appears on pair 29 (4th and 9th minutia in V_P) with key entry 4 which is indeed the target combination in our simulation. The final result of the attack is shown in Fig. 3(b).

Obviously, the result will better approximate the target vicinity if the search (phase) resolution is increased, which will also increase the search time. However, Table I suggests that already for a low resolution the correct key and minutiae pair can be identified. The depth of the resolution that yields a correct result is expected to depend on the target vicinity. As

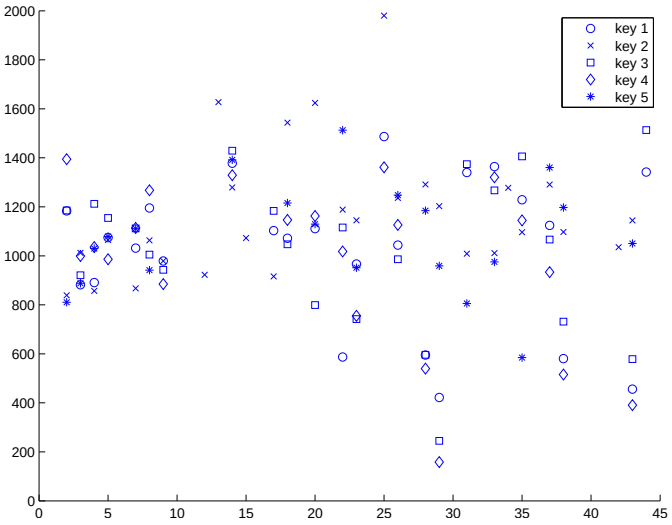


Fig. 4. The best distance error between the protected vicinity V_P and the regenerated vicinity V_P' for all 45 minutiae pairs and all 5 key entries. The phase resolution is $\theta_{res} = 30$. Combinations of pairs and key entries that are considered impossible have no distance error.

the phase estimate iterates from $-\pi/2$ to $\pi/2$ in $(2*\theta_{res}+1)$ steps, the complexity of the attack as given by (6) can be rewritten in term of the applied resolution

$$\binom{L*(M-1)}{M-1} \cdot L \cdot (2*\theta_{res}+1). \quad (7)$$

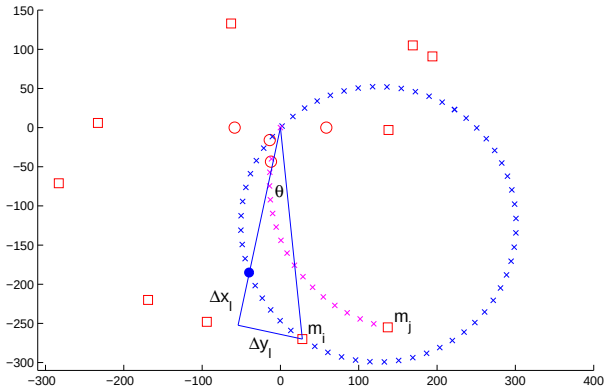
The best result in our table has a complexity of $45 \cdot 5 \cdot (2 \cdot 720 + 1) \approx 2^{18}$, which is way below an acceptable security level. The attack significantly improves the brute-force attack at the vicinity level and its running time is in the order of minutes. We note, however, that the lower resolutions, which run in seconds, correctly identify the pairs and the associated key entries. An adaptive algorithm that starts with a low resolution and maintains a list of candidate (minutiae pair, key entry) combinations could then adapt its resolution on focused regions to further reduce the complexity.

TABLE I
MINIMUM DISTANCE ERROR VERSUS PHASE RESOLUTION θ_{res}.

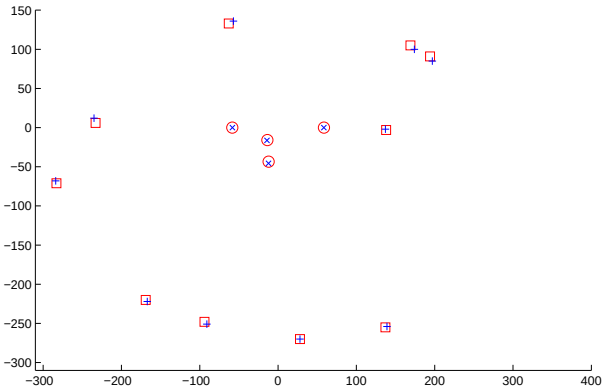
θ_{res}	e_{min}	Key	Pair	Complexity
30	158.2159	4	(4,9)	2^{14}
60	94.7099	4	(4,9)	2^{15}
90	42.0663	4	(4,9)	2^{15}
180	42.0663	4	(4,9)	2^{16}
360	42.0663	4	(4,9)	2^{17}
720	28.2448	4	(4,9)	2^{18}

B. Arbitrary Orientation (Dl) Attack

1) Attack Strategy: The Dmax attack exploits the opportunity to reduce the number of variables in the offset equations (1) when D_{max} is equal to D_l. While this reduces the complexity of the problem, it can easily be avoided by modifying the way the offsets are generated, e.g., by removing the D_{max} term or by using other orientations. We demonstrate



(a) Guessing trajectory of the Dmax attack for a given pair (m_i, m_j) and key entry (dx_l, dy_l). The red squares represent the protected minutiae in V_P. The red circles are the minutiae in V aligned along the longest orientation, before offsetting. The blue and magenta crosses show the trajectory of the recovered minutiae m'_i and m'_j, respectively, for the different guesses of θ. The phase resolution is $\theta_{res} = 30$. Invalid guesses are removed.



(b) Result of the Dmax attack with phase resolution $\theta_{res} = 90$. The chart includes the protected minutiae in V_P (red squares), the regenerated protected vicinity V'_P (blue plusses) for the best guess of θ, the original vicinity (red circles) and the cracked vicinity recovered with the Dmax algorithm (blue crosses).

Fig. 3. Visualization of the maximum orientation length (Dmax) attack.

in this section that we can equally well attack any other orientation O_l by iterating over all possible combinations of Δx_l and Δy_l.

Instead of testing all key entries we take one since any of the entries will have a minutiae pair corresponding to it. Our approach is similar to the Dmax attack. For a selected key entry we try all minutiae pairs in the protected vicinity and we exhaustively search the offsets Δx_l and Δy_l. Similarly as in the previous attack, we will set a search resolution and solve (1). Guesses are immediately eliminated if

- D_l or D_{max} is negative;
- D_l is larger than D_{max};
- D_{max} is not the length of the longest orientation in the recovered vicinity V'.

Each recovered vicinity V' is used to regenerate a protected vicinity V'_P, which is then compared with the given V_P. A distance error is computed and the guessed vicinity with the smallest error will indicate the pair that is associated with the selected key entry.

2) Algorithm: We do not guess Δy_l directly. As in the previous attack, we use the phase change after the offset. The resolution for this parameter is θ_{res}, the (distance) resolution for Δx_l is denoted by x_{res}. The range of the offset is limited by the boundaries of the template space. Let $x_{\max}$ denote the maximum distance from the origin. Hence, our guess for Δx_l goes from $-x_{max}$ to x_{max} with step x_{res}. The worst-case complexity of the attack is

$$\binom{L*(M-1)}{M-1} \cdot (2*\theta_{res}+1) \cdot \left(\left\lfloor \frac{2*x_{max}}{x_{res}} \right\rfloor + 1 \right) \quad (8)$$

The details of the attack algorithm are given in Appendix (Algorithm 2).

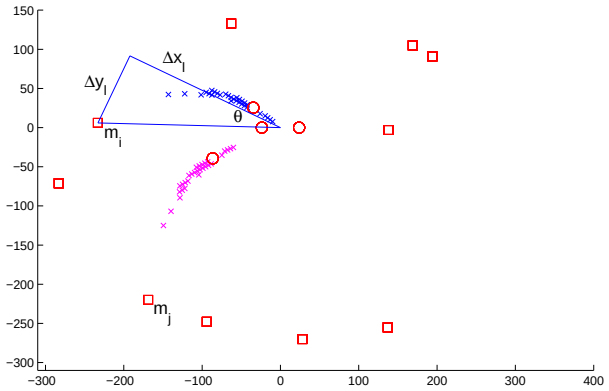
3) Experimental Result: The simulation is the same as in Section III-A3. A vicinity V_P is generated on which the Dl attack algorithm is run. Step 2 of the algorithm, i.e., the search for the best θ and $\Delta\tilde{x}_l$ given a minutiae pair (m_i, m_j) and an arbitrarily selected key entry (dx_a, dy_a) is shown in Fig. 5(a). As opposed to the Dmax attack, the potential pre-offset positions m'_i, m'_j do not lie on a single curve. Instead bands are formed because of the additional variable $\Delta\tilde{x}_l$ that has to be estimated. If the minutiae pair that was selected in V_P is correct for the selected key entry, then the two trajectories will approximate the two pre-offset minutiae that were aligned along the D_l orientation. The cracked vicinity produced by the Dl attack is shown in Fig. 5(b).

Similar conclusions as for the Dmax attack can be drawn for this attack. Table II presents the minimum distance error for given key entries and different values of the phase resolution and the distance resolution. Results that are incorrect, i.e., when the Dl attack associates the wrong minutiae pair with the given key entry, are denoted by /. The best (and completely correct) result is achieved with resolutions $\theta_{res} = 90$ and $x_{res} = 1$. If we approximate $2 * x_{max}$ in (8) with the height $H = 560$ of the template, the Dl attack has a worst-case complexity $45 \cdot (2 \cdot 90 + 1) \cdot (560 + 1) \approx 2^{22}$ for the given resolutions.

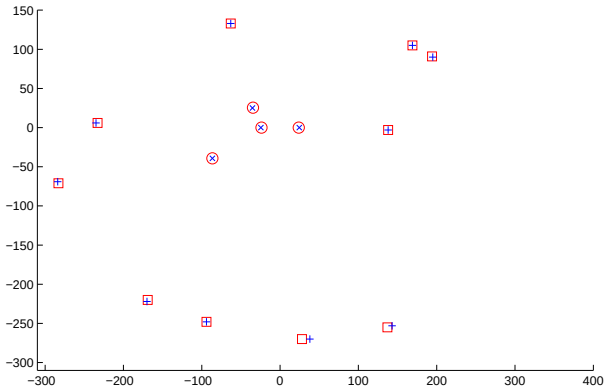
Table II suggests that a minimum search resolution is required to get a correct result for a given key entry. However, whereas the Dmax attack can match only one (unknown) key entry and minutiae pair, the Dl attack can be run against any given key entry individually. Individual runs at lower resolutions can be validated by looking at all runs as a whole to achieve a stronger result.

C. Full Fingerprint Example

To show the applicability of the algorithms, the Dl and Dmax attacks are applied on a sample from the FVC2000_DB4_B fingerprint database (Fig. 6(a)). In the



(a) Guessing trajectory of the Dl attack for a given pair (m_i, m_j) and arbitrarily selected key entry (dx_a, dy_a). Protected minutiae in V_P are represented by the red squares. The red circles are the target vicinity aligned along orientation O_l, before offsetting. The guessed minutiae m_i' and m_j' are given by the blue and magenta crosses for the different guesses of θ and $\Delta\tilde{x}_l$). The phase resolution is $\theta_{res} = 30$ and the Δx_l resolution is $x_{res} = 4$. Invalid guesses are removed.



(b) Result of the Dl attack with phase resolution $\theta_{res} = 90$ and Δx_l resolution $x_{res} = 2$. Red squares represent the protected minutiae in V_P, blue plusses are the regenerated protected vicinity V_P', red circles are the target vicinity and blue crosses are the cracked vicinity recovered with the Dl algorithm.

Fig. 5. Visualization of the arbitrary orientation (Dl) attack.

TABLE II
MINIMUM DISTANCE ERROR VERSUS PHASE AND $\Delta\tilde{x}$ RESOLUTION FOR DIFFERENT KEY ENTRIES. INCORRECT RESULTS ARE DENOTED BY /.

θ_{res}	x_{res}	key 1	key 2	key 3	key 4	key 5
30	4	/	687.0415	/	/	150.3345
30	2	/	235.5736	/	/	130.4024
30	1	/	224.3830	/	175.1013	130.4024
60	4	129.8887	143.3	/	/	115.2934
60	2	74.9108	143.2895	/	/	115.2934
60	1	57.6379	114.4178	/	121.0662	115.2934
90	4	112.0742	275.2957	/	56.6330	64.7344
90	2	26.7149	63.8929	46.8822	56.6330	64.7344
90	1	26.7149	63.8929	46.8822	56.6330	61.0708

sample fingerprint, 15 high quality minutiae are selected and used to generate 15 protected vicinities each with a random key table. The Dl and Dmax attacks are applied to each protected vicinity and the resolutions are set to $\theta_{res} = 60$ for both attacks and to $x_{res} = 1$ for the Dl attack. The Dmax attack has a lower complexity, whereas the Dl attack provides more flexibility or verification capabilities. There is, however, no significant difference in performance (in terms of e_{min}) between the two attack strategies. Given the key and the protected vicinities, the original vicinities can be inverted one by one. We note that by comparing the topological similarities between the reconstructed vicinities it is possible to gradually consolidate the original, non-vicinity based, minutiae template.

D. Increasing the Search Complexity

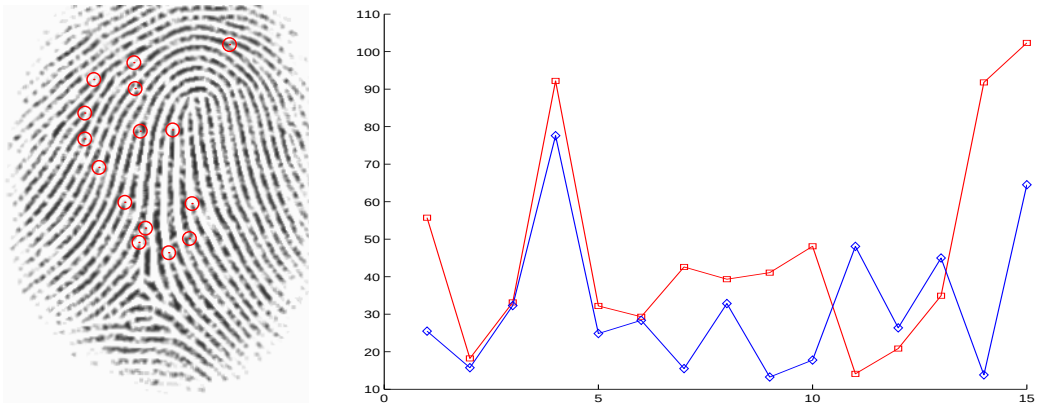
One of the main weak points in the attacked algorithm is that the offsets Δx_l and Δy_l depend on only a few variables. In combination with the vicinity approach this yields a relatively small search space for an attacker. A second weakness lies in the amount of information that is reused. The number of orientations that are used, the number of minutiae in a vicinity and the fact that some minutiae are offset more than once, give an attacker the means to verify the offsets that he estimates. This is most certainly the case for the orientation with maximum length ($D_l = D_{max}$).

A first improvement would be to no longer use the maximum length orientation for alignment. Secondly, the offset equations (1) should be modified to include more inherent biometric information and to introduce more non-linearity in the offsets. To make it further unsolvable, the offsets should be different for every aligned minutia and not be used for a pair of minutiae as is now the case. With these modifications, the search complexity should be increased to the level of a brute-force attack. However, despite the improvements the limited number of minutiae and the small template dimensions will remain a fundamental problem.

IV. CONCLUSION

In this paper we have analyzed the scheme of Yang and Busch for protecting fingerprint minutiae templates. Two algorithms were presented to attack and successfully reverse the protected templates. The attacks break the strong irreversibility of the method. This means that an attacker knows the key table K, which makes sense because it will probably be stored along with the protected templates. To break the weak irreversibility an attacker would have to guess the key table in the same way as the other variables were guessed in our attacks. The results seem to suggest that this might require significantly more efforts. What has to be analyzed is whether or not a unique solution still exists if the key has to be guessed as well.

Although this paper addresses one particular method which is focused on minutiae-based fingerprint templates, many of the ideas could potentially be applied to other types of cancelable biometrics algorithms that feature geometrical transformations. Because attack complexities are often overestimated, security criteria and evaluation methods for biometric template protection need to be further developed. We leave this and the



(a) Fingerprint with 15 high quality minutia points

(b) The red squares represent the e_{min} for the Dmax attack and the blue diamonds represent the e_{min} of the Dl attack, minimized over all key entries.

Fig. 6. Full fingerprint example: the Dl and Dmax attack applied on a protected fingerprint consisting of 15 minutiae vicinities.

extension of the presented attacks to other schemes for our future work.

REFERENCES

[1] S. Prabhakar, S. Pankanti, and A. K. Jain, "Biometric recognition: Security and privacy concerns," *IEEE Security and Privacy*, vol. 1, no. 2, pp. 33–42, March-April 2003.

[2] A. Cavoukian and A. Stoianov. (March 2007) Biometric encryption: A positive-sum technology that achieves strong authentication, security AND privacy. [Online]. Available: http://www.ipc.on.ca/

[3] G. Davida, Y. Frankel, and B. Matt, "On enabling secure applications through off-line biometric identification," *Proc. of the IEEE Symposium on Security and Privacy – S&P '98*, pp. 148–157, May 1998.

[4] A. Juels and M. Wattenberg, "A fuzzy commitment scheme," in *CCS '99: Proc. of the 6th ACM conference on Computer and Communications Security*, 1999, pp. 28–36.

[5] N. K. Ratha, J. H. Connell, and R. M. Bolle, "Enhancing security and privacy in biometrics-based authentication systems," *IBM Systems Journal*, vol. 40, no. 3, pp. 614–634, 2001.

[6] A. Juels and M. Sudan, "A fuzzy vault scheme," in *Proc. of IEEE International Symposium on Information Theory, Lausanne, Switzerland*, A. Lapidoth and E. Teletar, Eds. IEEE Press, 2002, p. 408.

[7] J.-P. M. G. Linnartz and P. Tuyls, "New shielding functions to enhance privacy and prevent misuse of biometric templates," in *AVBPA*, ser. LNCS, J. Kittler and M. S. Nixon, Eds., vol. 2688. Springer, 2003, pp. 393–402.

[8] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," in *Advances in Cryptology - EUROCRYPT 2004*, ser. LNCS, C. Cachin and J. Camenisch, Eds., vol. 3027. Springer, 2004, pp. 523–540.

[9] P. Tuyls, A. H. M. Akkermans, T. A. M. Kevenaar, G. J. Schrijen, A. M. Bazen, and R. N. J. Veldhuis, "Practical biometric authentication with template protection," in *AVBPA*, ser. LNCS, T. Kanade, A. K. Jain, and N. K. Ratha, Eds., vol. 3546. Springer, 2005, pp. 436–446.

[10] J. Bringer, H. Chabanne, G. Cohen, B. Kindarji, and G. Zémor, "Optimal iris fuzzy sketches," *First IEEE International Conference on Biometrics: Theory, Applications, and Systems, 2007. BTAS 2007.*, pp. 1–6, 27–29 September 2007.

[11] A. Beng Jin Teoh and C. T. Yuang, "Cancelable biometrics realization with multispace random projections," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, no. 5, pp. 1096–1106, 2007.

[12] J. Bringer, H. Chabanne, and B. Kindarji, "The best of both worlds: Applying secure sketches to cancelable biometrics," *Science of Computer Programming*, vol. 74, no. 1-2, pp. 43–51, 2008, special Issue on Security and Trust.

[13] R. Ang, R. Safavi-Naini, and L. McAven, "Cancelable key-based fingerprint templates," in *ACISP*, ser. LNCS, C. Boyd and J. M. G. Nieto, Eds., vol. 3574. Springer, 2005, pp. 242–252.

[14] U. Uludag, S. Pankanti, and A. K. Jain, "Fuzzy vault for fingerprints," in *AVBPA*, ser. LNCS, T. Kanade, A. K. Jain, and N. K. Ratha, Eds., vol. 3546. Springer, 2005, pp. 310–319.

[15] A. Arakala, J. Jeffers, and K. J. Horadam, "Fuzzy extractors for minutiae-based fingerprint authentication," in *ICB 2007*, ser. LNCS, S.-W. Lee and S. Z. Li, Eds., vol. 4642. Springer, 2007, pp. 760–769.

[16] T. Boult, W. Scheirer, and R. Woodworth, "Revocable fingerprint biotokens: Accuracy and security analysis," in *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR '07.*, IEEE. IEEE Computer Society, June 2007, pp. 1–8.

[17] E.-C. Chang and S. Roy, "Robust extraction of secret bits from minutiae," in *ICB 2007*, ser. LNCS, S.-W. Lee and S. Z. Li, Eds., vol. 4642. Springer, 2007, pp. 750–759.

[18] S. Draper, A. Khisti, E. Martinian, A. Vetro, and J. Yedidia, "Using distributed source coding to secure fingerprint biometrics," *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 2, pp. II–129–II–132, 15-20 April 2007.

[19] C. Lee, J.-Y. Choi, K.-A. Toh, and S. Lee, "Alignment-free cancelable fingerprint templates based on local minutiae information," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, no. 4, pp. 980–992, August 2007.

[20] N. K. Ratha, S. Chikkerur, J. H. Connell, and R. M. Bolle, "Generating cancelable fingerprint templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 4, pp. 561–572, 2007.

[21] B. Yang and C. Busch, "Parameterized geometric alignment for minutiae-based fingerprint template protection," in *IEEE 3rd International Conference on Biometrics: Theory, Applications, and Systems, 2009, BTAS '09*, 28–30 September 2009, pp. 1–6.

[22] F. Hao, R. Anderson, and J. Daugman, "Combining crypto with biometrics effectively," *IEEE Transactions on Computers*, vol. 55, no. 9, pp. 1081–1088, 2006.

[23] K. Simoens, P. Tuyls, and B. Preneel, "Privacy weaknesses in biometric sketches," in *2009 30th IEEE Symposium on Security and Privacy*, May 2009, pp. 188–203.

[24] A. Nagar, K. Nandakumar, and A. K. Jain, "Securing fingerprint template: Fuzzy vault with minutiae descriptors," in *Proc. International Conf. on Pattern Recognition (ICPR)*. IEEE, 2008, pp. 1–4.

[25] H. Xu, R. N. J. Veldhuis, A. M. Bazen, T. A. M. Kevenaar, T. A. H. M. Akkermans, and B. Gokberk, "Fingerprint verification using spectral minutiae representations," *Transaction on Information Forensics and Security*, vol. 4, no. 3, pp. 397–409, 2009.

[26] A. Nagar, K. Nandakumar, and A. K. Jain, "Biometric template transformation: a security analysis," *Media Forensics and Security II*, vol. 7541, no. 1, p. 75410O, 2010.

[27] L. Ballard, S. Kamara, and M. K. Reiter, "The practical subtleties of

biometric key generation," in *Proc. of the 17th Annual USENIX Security Symposium*, 2008, pp. 61–74.

[28] T. Kanade, A. K. Jain, and N. K. Ratha, Eds., *Audio- and Video-Based Biometric Person Authentication, 5th International Conference, AVBPA 2005, Hilton Rye Town, NY, USA, July 20-22, 2005, Proceedings*, ser. LNCS, vol. 3546. Springer, 2005.

[29] S.-W. Lee and S. Z. Li, Eds., *Advances in Biometrics, International Conference, ICB 2007, Seoul, Korea, August 27-29, 2007, Proceedings*, ser. LNCS, vol. 4642. Springer, 2007.

APPENDIX
ATTACK ALGORITHMS

Algorithm 1: Dmax attack for $M = 3$

Input: Protected vicinity $V_P = \{m_i\}_{i=1..2*L}$
Key table $K = \{(dx_l, dy_l)\}_{l=1..L}$
Phase resolution θ_{res}
Output: Vicinity $V = \{m'_i\}_{i=1..4}$
1. Iterate over all pairs in V_P and all key entries in K;
foreach *Pair (m_i, m_j) in V_P* **do**
  foreach *Key entry (dx_l, dy_l) in K* **do**
    Calculate the best guess for θ using step 2;
  end
end
2. Find the best θ given (m_i, m_j) and (dx_l, dy_l);
Let $e_{min} = \infty$ and $\theta_{min} = 0$;
for *$\theta = -\pi/2$ to $\pi/2$, step with $(\pi/2)/\theta_{res}$* **do**
  Let $\Delta\tilde{y}_l \leftarrow |m_i| \cdot sin(\theta)$;
  Compute $\Delta\tilde{x}_l$ from $\Delta\tilde{y}_l$ and (5);
  Reverse translate (m_i, m_j) to (m'_i, m'_j);
  `// Check validity of` m'_i `and` m'_j
  if *invalid guess* **then**
    Increase θ and skip rest of loop
  end
  Calculate D_l using (4);
  Let $m'_{x1} \leftarrow (-D_l/2, 0)$ and $m'_{x2} \leftarrow (D_l/2, 0)$;
  Regenerate V'_P from $\{m'_i, m'_j, m'_{x1}, m'_{x2}\}$;
  `// Calculate similarity` V_P `and` $V_P{}'$
  foreach m'_i *in V'_P* **do**
    Find closest m_i in V_P;
    Calculate distance $d_i = d(m_i, m'_i)$;
    Remove m_i and m'_i from V_P and V'_P, respectively;
  end
  `// Update best guess`
  if $\sum d_i < e_{min}$ **then**
    Let $e_{min} = \sum d_i$ and $\theta_{min} = \theta$
  end
end
3. Create performance list L with entries $\{(m_i, m_j), (dx_l, dy_l), e_{min}, \theta_{min}\}$. Each entry indicates the best distance error which is obtained for the given minutiae pair and key entry by by using the estimate θ;
4. Select entry in L with smallest e_{min} to recover unprotected vicinity V ;
Return: V

Algorithm 2: Dl attack for $M = 3$

Input: Protected vicinity $V_P = \{m_i\}_{i=1..2*L}$
Key table $K = \{(dx_l, dy_l)\}_{l=1..L}$
Phase resolution θ_{res} and distance resolution x_{res}
Output: Vicinity $V = \{m'_i\}_{i=1..4}$
1. Select an arbitrary key entry (dx_a, dy_a) from K ;
foreach *Pair (m_i, m_j) in V_P* **do**
  Calculate the best guess for θ and $\Delta\tilde{x}_l$ using step 2;
end
2. Find the best θ and $\Delta\tilde{x}_l$ given (m_i, m_j) and (dx_a, dy_a);
for *$\theta = -\pi/2$ to $\pi/2$, step with $(\pi/2)/\theta_{res}$* **do**
  for *$\Delta\tilde{x}_l = -x_{max}$ to x_{max}, step with x_{res}* **do**
    Let $\Delta\tilde{y}_l \leftarrow |m_i| \cdot sin(\theta)$;
    Compute D_l and D_{max} from $\Delta\tilde{y}_l$, $\Delta\tilde{x}_l$, key entry (dx_a, dy_a) and (5);
    `// Check validity` D_l `and` D_{max}
    if *$D_l < 0$ or $D_{max} \leq 0$ or $D_{max} < D_l$* **then**
      Increase $\Delta\tilde{x}_l$ and skip rest of loop;
    end
    Reverse translate (m_i, m_j) to (m'_i, m'_j);
    `// Check validity of` m'_i `and` m'_j
    if *invalid guess* **then**
      Increase $\Delta\tilde{x}_l$ and skip rest of loop;
    end
    Let $m'_{x1} \leftarrow (-D_l/2, 0)$ and $m'_{x2} \leftarrow (D_l/2, 0)$;
    `// Check validity of` D_{max} `in` V'
    if *$D_{max} \notin O_l(V')$ or $D_{max} \neq \max(O_l(V'))$* **then**
      Increase $\Delta\tilde{x}_l$ and skip rest of loop;
    end
    Regenerate V'_P from $V' = \{m'_i, m'_j, m'_{x1}, m'_{x2}\}$;
    `// Check similarity` V_P `and` $V_P{}'$
    foreach m'_i *in V'_P* **do**
      Find closest m_i in V_P;
      Calculate distance $d_i = d(m_i, m'_i)$;
      Remove m_i and m'_i from V_P and V'_P;
    end
    `// Update best guess`
    if $\sum d_i < e_{min}$ **then**
      Let $e_{min} = \sum d_i$, $\theta_{min} = \theta$ and $\Delta\tilde{x}_{min} = \Delta\tilde{x}_l$
    end
  end
end
3. Idem as Dmax, except: add $\Delta\tilde{x}_{min}$ to entries
4. Idem as Dmax;
Return: V